BPR: Blockchain-Enabled Efficient and Secure Parking Reservation Framework With Block Size Dynamic Adjustment Method

Jishu Wang, *Member, IEEE*, Chao Zhu, Chen Miao, Rui Zhu, *Member, IEEE*, Xuan Zhang, *Member, IEEE*, Yahui Tang, Hexiang Huang, and Chen Gao, *Graduate Student Member, IEEE*

***Abstract*— The parking lot is one of the important components of the intelligent transportation system (ITS). The current parking lots mainly use instant parking, which has low parking efficiency, during peak hours, which leads to traffic congestion. To guarantee the stable operation of parking lots, we propose a blockchain-enabled parking reservation framework, called BPR. Traditional parking reservation systems may exist the condition of malicious reservations, and resulting in wasted parking spaces. Therefore, we design a reputation mechanism to manage the parking reservation behavior of vehicles and reduce the number of malicious nodes. In addition, to balance the performance of the blockchain at different times (especially during peak hours), we use deep learning (DL) to dynamically adjust the block size to make the blockchain run more efficiently and stably. We deploy the system in Hyperledger Fabric and conduct effectiveness experiments. The comprehensive evaluation results and analysis show that the proposed reputation mechanism can effectively curb malicious nodes from reserving parking spaces and reduce the waste of parking resources. And the block size will be dynamically adjusted to balance the performance of the blockchain at different periods, this method is also applicable to other blockchain performance-sensitive scenes. Finally, this paper is compared with related work to demonstrate the innovation and feasibility of this work from various aspects.**

***Index Terms*— Blockchain, parking reservation, deep learning, reputation, block size.**

Manuscript received 3 July 2022; revised 12 October 2022; accepted 7 November 2022. Date of publication 23 November 2022; date of current version 1 March 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62002310, Grant 61262025, Grant 61502413, and Grant 61862063; in part by the Yunnan Science and Technology Major Project under Grant 202002AD080002; in part by the Yunnan Provincial Natural Science Foundation Fundamental Research Project under Grant 202101AT070004; in part by the Yunnan Province Software Engineering Key Laboratory Open Fund Project under Grant 2020SE301 and Grant 2020SE404; in part by the National Social Science Foundation of China under Grant 18BJL104; in part by the Science Foundation of Young and Middle-Aged Academic and Technical Leaders of Yunnan under Grant 202205AC160040; in part by the Science Foundation of Yunnan Jinzhi Expert Workstation under Grant 202205AF150006; in part by the Science and Technology Innovation Project of Yunnan Provincial Transportation Investment and Construction Group Company under Grant YCIC-YF-2021-07; in part by the Yunnan Xing Dian Talents Support Plan; and in part by the 14th Postgraduate Research Innovation Project of Yunnan University under Grant KC-2222958. The Associate Editor for this article was C. Lv. *(Corresponding authors: Rui Zhu; Xuan Zhang.)*

Jishu Wang and Chen Gao are with the School of Information Science and Engineering, Yunnan University, Kunming 650091, China (e-mail: cswangjishu@hotmail.com; 929165733@qq.com).

Chao Zhu, Chen Miao, Rui Zhu, Xuan Zhang, and Hexiang Huang are with the School of Software, Yunnan University, Kunming 650091, China (e-mail: 704026774@qq.com; 965378706@qq.com; rzhu@ynu.edu.cn; zhxuan@ynu.edu.cn; roland@mail.ynu.edu.cn).

Yahui Tang is with the School of Software, Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: 457923156@qq.com).

Digital Object Identifier 10.1109/TITS.2022.3222960

I. Introduction

THE parking lot play an important role in ITS, and every city has many parking lots. They can be divided into free parking lots and paid parking lots by charging mode; public parking lots and private parking lots by function. The distribution and number of parking lots may affect the traffic congestion level of cities [1], [2].

However, with the increase in car ownership and the construction of urban roads, the parking lots at this stage often have difficulty meeting the demand, especially during the peak hours. For drivers, finding parking spaces is both time-consuming and laborious [3]. Solving such problems by adding more parking spaces alone would require significant economic costs and would require modifications to existing urban planning, which may further complicate the problem.

Drivers are often left to search blindly, which is one of the reasons for parking congestion. According to a 2017 USA Today report, the average U.S. driver spends 17 hours per year searching for a parking space on a street, in a lot, or in a garage. In New York City, the hardest-hit metropolitan area in the U.S., drivers spend an average of 107 hours per year searching for a parking space. The report estimated the cost of wasted time, fuel, and emissions at $2,243 per driver [4].

At present, many studies have focused on solving the problem of parking space navigation (e.g., improved navigation algorithm [5], [6]), parking schemes (e.g., parking management system [7], [8], parking guidance [9], [10], [11], parking assignment [12], [13]), and traffic flow pridiction [14].

Enabling parking reservation is a good way to this problem. Studies have shown that the average parking-related traffic during peak hours can be 30-50% of the total traffic volume [15]. Therefore, parking reservation will play an important role in relieving urban traffic congestion (especially during peak hours). Currently, some studies have focused on parking reservation solutions [16], [17], [18]. However, few researchers have considered the reputation of vehicles in these solutions. Also, when conducting parking reservation, the vehicle needs

to provide some private information, so attention to the privacy of the vehicle is also needed.

Based on the immutability and traceability of blockchain, some solutions for parking reservations can be designed in combination with reputation mechanisms. Currently, the application of blockchain technology to the Internet of Vehicles (IoV) has received some attention [19]. To solve the problems in the IoV, some researchers have proposed different schemes for the combination of the IoV and the blockchain (e.g., privacy protection [20], [21], [22], [23], [24], vehicle life cycle [25], vehicle supply chain [26], vehicular edge computing [23], [27], [28], [29], electronic toll collection [30]. However, these existing blockchain-based solutions do not directly address the problems of parking systems. Limited by the number of vehicles and the performance of onboard devices, the consortium chain platform is more suitable than the public chain platform for the problems we want to solve.

Specifically, we propose a blockchain-based parking system framework to build communication between vehicles and parking lots for protecting drivers' private information and making parking more efficient. We design an algorithm to simulate based-reservation parking mode. At the same time, we also design a reputation mechanism to manage the reservation behavior of vehicles to prevent malicious vehicles from reserving unlimited parking spaces and wasting of parking resources. However, the introduction of blockchain leads to some performance bottlenecks (during peak hours, this can lead to higher latency or lower throughput, which can hinder blockchain network operations), because block size (usually set artificially) and transaction arrival rate (parking transaction flow in this paper) can significantly affect the performance of blockchain, and the parking transaction flow at different time slots is different in real-world scenes. Therefore, we introduce DL methods to dynamically adjust the block size to balance the performance (throughput and latency) of the blockchain under different traffic flow to meet the requirements of real-world scenes, to the best of our knowledge, this is the first research work that uses DL methods to dynamically adjust the block size.

The main contributions of this paper are described as follows.

1) We propose a blockchain-based parking system framework that is more efficient and secure than traditional parking systems, namely BPR. In this framework, we design a blockchain-based parking architecture that can effectively protect the privacy of drivers and reserve parking. To prevent the waste of parking resources due to unlimited reservations by malicious vehicles, we design a reputation mechanism to manage the parking behavior of vehicles.
2) To balance the performance of the blockchain under different periods, we introduce DL methods to dynamically adjust the block size. To the best of our knowledge, this is the first time that DL methods is used to dynamically adjust the block size and balance the performance of blockchain. This method can be used not only for the parking lot, but also for other performance-sensitive blockchain scenes (e.g., online shopping mall, bank credentials). In addition, we also contribute a real-world blockchain performance dataset for other researchers to use.
3) We deploy the proposed system. The experimental results and analysis show that, our proposed parking system framework is higher scalability compared to existing related work. Because the reservation-based parking mode is more efficient than the traditional parking mode. The reputation-based vehicle parking behavior management can effectively prevent malicious vehicles from making failed reservations. The DL-based block size dynamic adjustment method can effectively adjust the block size in different periods, so that the performance of the blockchain can be balanced in different periods.

The rest of this paper is organized as follows: in Section II, we discuss the work related to parking reservation, blockchain-based parking lot systems, and DL with blockchain. In Section III, we introduce BPR system model, including the architecture design of BPR, reservation-based parking mode, and a reputation-based mechanism for managing vehicle parking behavior. Section IV then presents the details of the DL-based block size dynamic adjustment method. Section V describes the deployment of our proposed system. In Section VI, we evaluate and analyze the performance and features of our proposed system. Finally, Section VII concludes the paper.

II. Related Work

A. Parking Reservation

In recent years, researchers have put forward some schemes for making reservations for parking. For instance, Tasseron et al. [31] used an agent-based analysis method to study the impact of the on-street parking reservation system and proved that parking reservations can effectively reduce the time for drivers to find parking spaces, but they did not consider the impact of unlimited reservations on parking spaces. Wan et al. [32] proposed a secure crowdsourcing-based parking reservation system called SCPR, private owners can rent out parking spaces. Huang et al. [18] proposed a privacy-preserving reservation scheme for autonomous vehicles to protect the private information of drivers. To avoid repeated reservations for vehicles during the reservation period, they set up a reservation token, but they also did not consider those malicious vehicles may make unlimited parking reservations outside of the scheduled time. Waheed et al. [17] proposed a learning automaton and reservation-based secure smart parking system, which effectively reduces the time for drivers to search for parking spaces. But they did not manage the parking reservation behavior of vehicles.

It is important to manage the parking reservation behavior of vehicles. However, the existing research work has studied this part very little. This is the motivation for us to design the reputation-based mechanism and introduce blockchain.

B. Parking Lot System Based on Blockchain

Blockchain has many applications in the field of ITS. The combination of blockchain and the parking systems has also

attracted the attention of some researchers. Hu et al. [33] proposed a blockchain-based privacy-preserving system, which aims to protect the privacy of users without relying on third-party entities. Zhang et al. [34] proposed a blockchain-enabled smart parking scheme called BSFP, which efficiently realized the reliability and fairness of parking and the privacy protection of private parking space owners. Wang et al. [35] proposed a rewards Airbnb-like privacy-enhanced private parking spot sharing scheme based on blockchain, this solution does not require the participation of a trusted third party and realizes security and privacy protection under the premise of acceptable computing cost and communication overhead. Badr et al. [36] presented a smart parking system with privacy preservation and reputation management using blockchain, in this system, drivers can evaluate parking services anonymously to ensure high-quality services. Zhu et al. [14] proposed an anonymous smart parking and payment scheme in an in-vehicle network. They used short random signatures to provide anonymity and conditional privacy. Also they used hashmap for fast result matching and E-cash for anonymous payments. Li et al. [24] proposed an efficient and privacy-preserving parking-space recommendation service platform, namely PriParkRec, along with the proof-of-concept solution to protect the requester's privacy.

Inspired by the above work, we propose a parking system framework based on blockchain and reputation mechanism to improve some shortcomings of the current work and make it more feasible and applicable.

C. DL With Blockchain

Both DL and blockchain are currently popular research directions. Therefore, research on the combination of DL and blockchain has also received a lot of attention.

To address some of the challenges in service request orchestration in software-defined networks (SDN), Zhang et al. [37] proposed a solution based on artificial intelligence and blockchain. It is demonstrated theoretically and empirically that the proposed algorithm can provide efficient and intelligent request orchestration in SDN with extreme security. For the 5G intelligent Internet of Things, Rathore et al. [38] proposed a deep learning and blockchain authorization security framework, which uses DL capabilities for inteligent data analysis operations and uses blockchain to protect data security. Wang et al. [39] proposed a blockchain-based in-vehicle crowd perception system to protect user privacy and data security in 5G IoV and to maximize security and minimize blockchain delays, an algorithm that enables deep reinforcement learning (DRL) is proposed to select suitable active miners and transactions. To detect driver behavior in smart cars, Khan et al. [40] proposed a DL and blockchain fusion solution, experiments with related work show that this method has a high accuracy rate on the test data set. Dai et al. [41] proposed a solution based on deep reinforcement learning and consortium blockchain to implement secure network content caching through in-vehicle edge computing. Boateng et al. [42] proposed a novel hierarchical framework and a multi-agent DRL method for blockchain-empowered spectrum trading for network slicing in radio access network. Security assessment and extensive simulation results confirmed the security and efficiency of this proposed method in terms of players' utility maximization and fairness, compared with other baselines. Boateng et al. [43] proposed a hierarchical blockchain-DRL framework for secure resource trading and autonomous resource slicing in 5G radio access network, and they deployed a consortium blockchain network that supports hyperledger trading smart contract to ensure security and transparency in resource trading between a single buyer and multiple seller service providers.

As far as we know, no researcher has used DL methods to dynamically adjust block size in the current research work. However, throughput and latency are the most important performance metrics in blockchain, which in turn is the most important motivation for our work.

III. BPR System Model

In this section, we will provide an overview of the proposed system model and introduce the details of BPR from three aspects: architecture design, reservation-based parking mode, and vehicle parking behavior management based on reputation mechanism.

A. Architecture Design

In BPR system architecture, there are mainly four types of nodes: Certificate Authority (CA), Road Side Unit (RSU), On-Board Unit (OBU), and Server, as shown in Fig. 1.

1) CA: CA certifies the validity of each node, issue, and revoke certificates. Before each node joins the blockchain network, it needs to register with the CA and obtain a certificate.
2) RSU: RSU is responsible for communicating with vehicles and servers, and transmitting parking requests and results.
3) OBU: OBU (vehicle) is responsible for communicating with RSU, initiating parking requests, and receiving parking results. OBU play the role of a Client. In Hyperledger Fabric, the Client can initiate transactions and invoke the chaincode to query its own information, but it does not need to participate in the maintenance of the blockchain (e.g., sorting, verification, packaging).
4) Server: The server is responsible for communicating with the RSU, receiving parking requests, and returning the parking result according to the real-time parking situation of the parking lot. As a full node, the server is responsible for the ordering, packaging, and storage of ledgers in the blockchain. In realistic scenarios, clusters of multiple servers are often used to maintain the blockchain network.

Because the traditional public chain architecture requires many nodes to participate, it not only requires nodes to have sufficient computing power but also requires nodes to have sufficient storage space. Considering the performance of in-vehicle equipment and the impact of the number of nodes on network performance, BPR uses the consortium chain as the blockchain framework. Among the current consortium chain

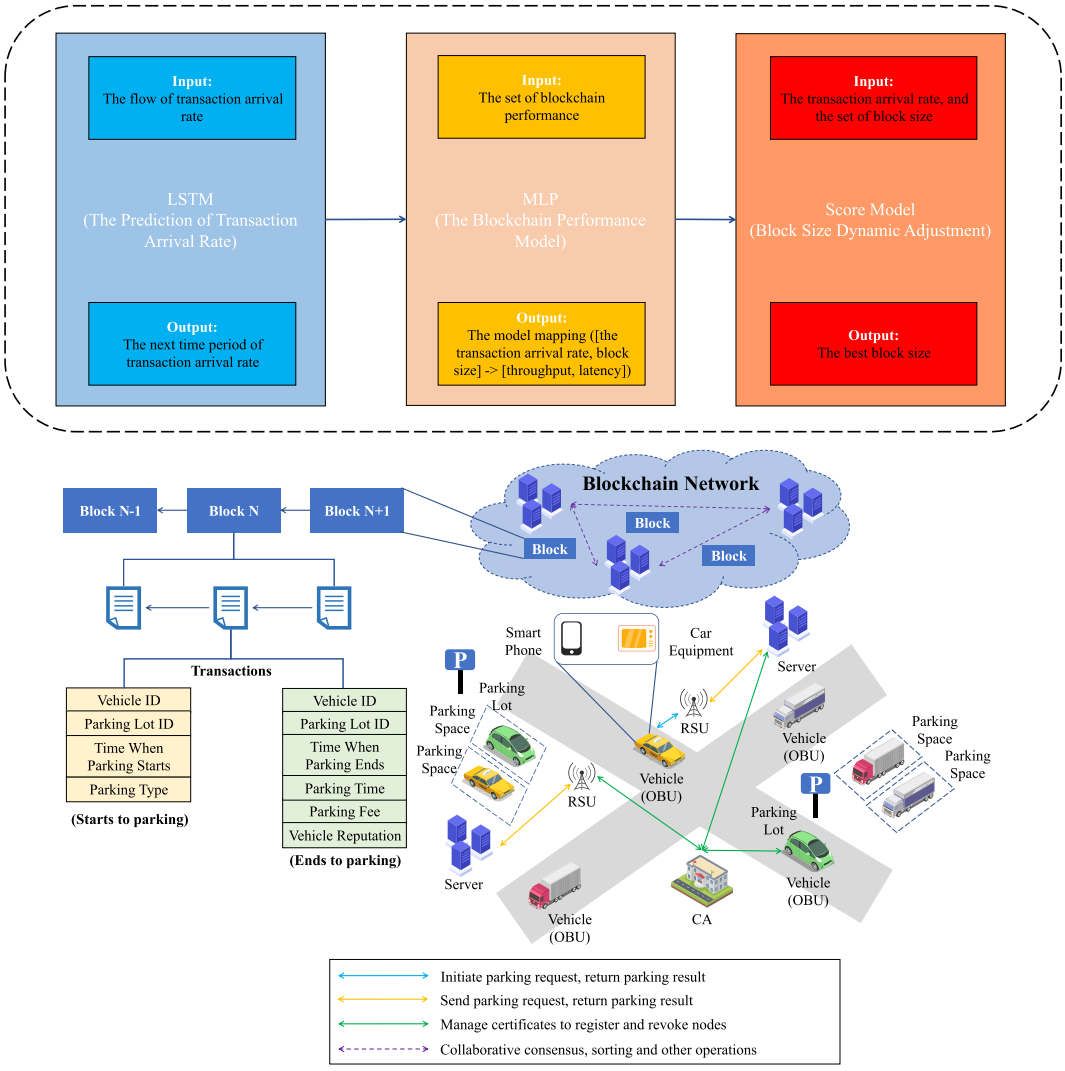


Fig. 1. BPR system model.

platforms, Hyperledger Fabric is one of the most popular platforms. Due to its channel, organization, and high throughput characteristics, it is more suitable for parking lot problem scenes.

To better record, the information on vehicle parking, ensure the efficiency of vehicle parking, and prevent malicious drivers from initiating unlimited reservations to occupy system resources and deny their malicious behavior. We store the information when the vehicle initiates a parking reservation and the information when the vehicle ends parking as a transaction in the block, the blockchain structure is shown in Fig. 1. Drivers can use the reservation mode for parking, or they can use the regular parking mode (i.e., arrive directly at the parking lot to park, but this mode does not ensure the availability of parking spaces). In BPR, the number of each parking space is not limited. In practice, parking managers can set the number of parking spaces according to the actual demand, and maximize the parking efficiency.

In BPR, the specific process for the driver to complete parking is shown in Fig. 2. All nodes in the blockchain need to register with the CA, and then obtain a certificate. The driver

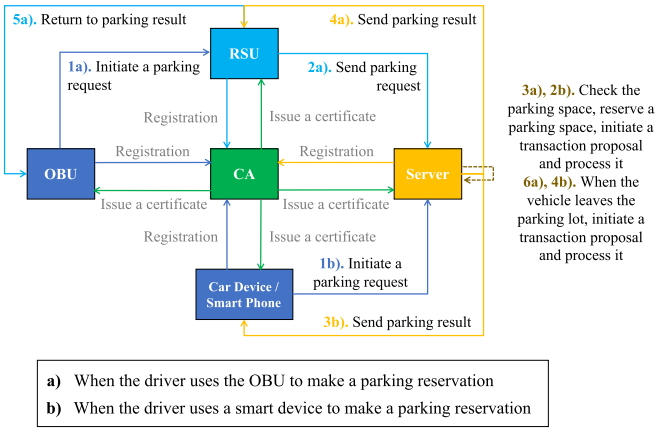


Fig. 2. A complete parking process in BPR.

can not only initiate a parking reservation request through the OBU but also initiate the request through the in-vehicle device or smartphone.

1) When the driver uses the OBU to make a parking reservation, a total of 6 steps are required. In step 1, the OBU initiates a parking reservation request to the RSU. In step 2, RSU forwards the request to the server. In step 3, the server checks the parking space situation of the parking lot currently, if there are remaining parking spaces, it reserves a parking space for the vehicle, and initiates and processes the requested information as a transaction in the blockchain. In step 4, the server sends the parking result to the RSU. In step 5, the RSU returns the parking result to the OBU. In step 6, when the vehicle leaves the parking lot, the parking lot will charge the parking fee to the vehicle (no charge if the parking lot is free), and the server will initiate and process the parking information as a transaction in the blockchain.
2) When a driver uses a smart device to make a parking reservation, it can be divided into 4 steps. In step 1, the driver initiates a parking request to the server through the smart device. In step 2, the server checks the parking space situation of the parking lot currently, if there are remaining parking spaces, it reserves a parking space for the vehicle, and initiates and processes the requested information as a transaction in the blockchain. In step 3, the server sends the parking result to the smart device. In step 4, when the vehicle leaves the parking lot, the parking lot will charge the parking fee to the vehicle (no charge if the parking lot is free), and the server will initiate and process the parking information as a transaction in the blockchain.

Because the server has sufficient computing power, the main operations (e.g., transaction initiation, sorting, verification, and packaging) in the blockchain are performed by the clusters of multiple servers. With the development and popularization of smart devices, more and more applications are being carried on smart devices [44]. Therefore, in the future, researchers can design some parking reservation applications and carry them on smart devices.

B. Reservation-Based Parking Mode

In this part, we present a algorithm that simulate and describe real-world driver for reserving parking: the reservation-based parking mode. The notations used in this section are illustrated in Table I.

Algorithm 1 is the pseudo-code of the reserved parking mode algorithm. We divide the execution flow of Algorithm 1 into 3 steps, details are given as follows.

1) Initialize: Initialize the variables, as shown in lines 2-6 of the algorithm.
2) Choose a parking lot: Choose a parking lot that best suits and make a reservation, as shown in lines 9-14 of the algorithm.
3) Parking: Go to the parking lot of the choice and complete the parking operation, as shown in lines 15-16 of the algorithm.

Compared to the regular parking mode, the reservation-based parking mode improves parking efficiency in two aspects, and reduces the impact of finding parking spaces on urban traffic congestion.

TABLE I
LIST OF NOTATIONS

Notation	Description
$LoV = \{Lon_{Cur}, Lat_{Cur}\}$	The location of the vehicle
Lon_{Cur}	The current longitude of the vehicle
Lat_{Cur}	The current latitude of the vehicle
$PL = \{P_1, ..., P_n\}$	The set of parking lots
$P = \{Lon, Lat\}$	The parking lot in PL
P_N	The nearest parking lot to the vehicle
Dis_{Cur}	Distance between the LoV and the P
R_{max}	Maximum reservation distance
S_{ave}	The average speed of the vehicle
T_B	The time when the vehicle begins to parking
T_D	The duration of the vehicle parking
T_E	The time when the vehicle ends parking
R_i	The reputation value of node i
R_i^I	The initial R_i
R_i^S	The success behaviors part of R_i
R_i^F	The failure behaviors part of R_i
NS_i	The number of success behaviors of node i
NF_i	The number of failure behaviors of node i
NPF_i^j	The number of free parking spaces when node i performs the j_{th} failure behavior
C_i^j	The capacity of the parking lot when node i performs the j_{th} failure behavior
TS_i^j	The duration of the j_{th} success behavior of node i
TF_i^j	The time of the j_{th} failure behavior of node i (duration of the reservation status)
TP_i	The penalty suffered by node i after conduct the j_{th} failure behavior
PF_i	The penalty coefficient corresponding to the j_{th} failure behavior of node i
λ	The parking reservation restriction coefficient
β_1, β_2, β_3	The penalty coefficient in different situations
ω	The number of consecutive NF_i

1) The driver can check the parking space of each parking lot at any time, and choose a target parking lot to reserve parking, without having to reach the specific parking lot, which reduces the time to find the parking spaces in the parking lot one by one.
2) After the driver makes a parking space reservation in a specific parking lot, the parking time starts to be calculated, instead of starting to calculate the parking time after arriving at the parking lot, which can improve the utilization rate of the parking lot.

The time complexity of Algorithm 1 mainly depends on two points: 1) The parking lot finding range set by the driver; 2) The number of qualified parking lots in that range. In reality, drivers tend to look for closer parking lots rather than farther ones, so the number of compliant parking lots will not be too large. The time complexity of Algorithm 1 is $O(n)$.

C. Vehicle Parking Behavior Management Based on Reputation Mechanism

After enabling reserved parking, the driver's parking efficiency will be improved, and the traffic congestion on urban roads will also be reduced, but many problems will also accompany, the most important of which is malicious nodes unrestricted parking space reservations may be made, but

Algorithm 1 Reservation-Based Parking Mode

Input:
Latitude and longitude of vehicle position: LoV;
The set of parking lots: PL;
Parking start time: T_B;
Duration of vehicle parking: T_D;
The average speed of the vehicle: S_{ave};
Procedures:
    $Dis_{Cur} = R_{max}$
    foreach$(P : PL)$
        if Distance$(LoV, P) < Dis_{Cur}$ **then**
            Dis_{Cur} =**Distance**(LoV, P)
            $P_N = P$
        end if
    end foreach
    Reserve a parking space and proceed to parking;
    $T_E = T_B + T_D$

parking is not performed, resulting in idle parking space resources and waste of network computing resources.

Currently, some methods can be used to curb malicious reservations.

1) Limiting the duration of the reservation status (e.g., the reservation status only lasts 1 minute).
2) Limiting the number of reservations (e.g., only two reservations per day).

However, these methods are not suitable for parking lot scenes, because drivers often need to make parking reservations dozens of minutes in advance to enable them to have enough time to get to the parking lot, and many drivers need to park multiple times in a day. Aiming at the shortcomings of the current method, we design a management mechanism for vehicle reservation behavior based on reputation value.

The principles we consider mainly consist of 3 parts.

1) For honest nodes that occasionally fail to reserve parking (e.g., occasionally canceling the parking, changing the parking lot, and failing to arrive at the parking lot within the specified time), the reputation value changes less.
2) For malicious nodes that frequently make reservations and fail to park (e.g., make parking reservations continuously or frequently, but rarely or never actually park in the parking lot), their reputation value changes greatly. And for such nodes, will be punished to prevent them from making unlimited reservation behaviors.
3) Malicious nodes often begin to conduct malicious behaviors frequently after joining the network, instead of performing malicious behaviors after a series of honest behaviors (this is because it requires it to pay a significant cost upfront before engaging in malicious behavior).

Especially in the parking lot that requires a fee, the driver needs to pay some parking fees after every successful parking. This also avoids the situation where honest nodes suddenly turn into malicious nodes (frequent failure behaviors), because this situation requires certain costs.

The symbols used in this part are shown in Table I. The behavior of the node is divided into success behavior (successful parking) and failure behavior (active cancellation of reservation or reservation timeout). Therefore, take node i as an example, R_i is determined by three parts, which can be expressed as

$$R_i = R_i^I + R_i^S - R_i^F \tag{1}$$

R_i is affected by both R_i^I, R_i^S and R_i^F. R_i^I represents the initial reputation value. Setting a reasonable R_i^I can effectively prevent a vehicle from becoming a malicious node when it cancels its reservation due to unforeseen circumstances when it first uses the reservation for parking.

To manage nodes more efficiently, we divide them into two categories.

1) *Honest Node*: For this type of node, even if it occasionally conducts failure behavior, its reputation value will not be greatly changed. The judgment standard is: the number of consecutive failure behaviors $< \omega$, and $R_i \geq 0$.
2) *Malicious Node*: For such nodes, the change in their reputation value will be more drastic and will be penalized by restricting reservations. The judgment standard is: the number of consecutive failure behaviors $\geq \omega$, or $R_i < 0$.

R_i^S is calculated from each success behavior and can be expressed as

$$R_i^S = \sum_{j=1}^{NS_i} TS_i^j \tag{2}$$

R_i^S is determined by the number of success parking of node i and the length of each parking time (the unit is minutes). The longer the parking time, the faster R_i^S will increase, which also corresponds to the situation in the real scene (the longer the parking time, the higher the cost). Once node i becomes a malicious node, R_i^S will be reset to 0, and every time it fails, it will cause R_i^S to be reset to 0, until it performs the success behavior or becomes an honest node, R_i^S will continue increase.

R_i^F is calculated by each failure parking behavior of node i, which can be expressed as

$$R_i^F = \begin{cases} \sum_{j=1}^{NF_i} j \cdot TF_i^j \cdot PF_i^j, & \text{Malicious Node;} \\ \sum_{j=1}^{NF_i} TF_i^j \cdot PF_i^j, & \text{Honest Node.} \end{cases} \tag{3}$$

R_i^F is composed of multiple parts, including the number of failure behaviors performed by node i, the reservation duration corresponding to each failure behavior, and the penalty coefficient corresponding to each failure behavior. Obviously, the greater the number of failure behaviors, the greater j, which will cause R_i^F to increase faster and faster.

In different parking situations, the penalty coefficient PF_i^j is different. Which can be expressed as

$$PF_i^j = \begin{cases} \beta_1, & \text{if } \frac{NPF_i^j}{C_i^j} <= 0.1; \\ \beta_2, & \text{if } 0.1 < \frac{NPF_i^j}{C_i^j} <= 0.5; \\ \beta_3, & \text{otherwise.} \end{cases} \tag{4}$$

especially, malicious behavior during peak hours (when there are fewer parking spaces) is a worse situation. Therefore, we can set different parameters, and effectively prevent malicious behavior from occurring during peak hours.

The unlimited reservation of parking by nodes will waste parking space resources and network resources. We will restrict the function of parking reservations based on the node's reputation value to prevent this from happening. TP_i is the duration for which node i is restricted from using the parking reservation, which can be expressed as

$$TP_i = \begin{cases} \frac{R_i}{\lambda}, & \text{Malicious Node;} \\ 0, & \text{Honest Node.} \end{cases} \tag{5}$$

λ is the restriction coefficient of parking reservation. For example, when R_i^F is 1000, if we set λ to 20, then node i will not be able to use the parking reservation within 50 seconds. Therefore, by adjusting λ, malicious reservations can be effectively curbed.

All in all, in this reputation mechanism, nodes that frequently conduct malicious behaviors will be punished more and more severely. This reputation mechanism can effectively curb the occurrence of malicious reservations while ensuring that honest nodes use parking reservations.

IV. Dynamic Adjustment Method of Block Size

In this part, we dynamically adjust the block size based on DL methods to balance the performance (throughput and latency) of the blockchain.

A. The Prediction of Transaction Arrival Rate

First of all, we predict the number of short-term transaction arrival rate in the blockchain based on Long Short-Term Memory (LSTM) [45]. Then we model the effect of transaction arrival rate, block size on latency, and throughput respectively based on Multilayer Perceptron (MLP). Finally, we set a custom score function as an evaluation metric to trade off latency and throughput. In practice, we can dynamically adjust the block size according to the changing number of transaction arrival rate to achieve the best evaluation metric, and the scoring function can be also adjusted according to actual performance needs.

Predicting the number of short-term transaction arrival rate in the parking lot is a typical time-series forecasting problem which is to predict the most likely number of transaction arrival rate in the next P time steps given the previous T observations, which can be expressed as

$$\hat{x}_{t+1}, \ldots, \hat{x}_{t+P} = \underset{x_{t+1}, \ldots, x_{t+P}}{\operatorname{argmax}} \log P(x_{t+1}, x_{t+P} | x_{t-T+1}, \ldots, x_t) \tag{6}$$

where x_t is the number of transaction arrival rate at time step t. In this work, we employ 60 minutes as the historical time window, also known as 12 observed data points ($T = 12$) are used to forecast the transaction arrival rate in the next 5 minutes ($P = 1$). We set the frequency of prediction to 5 minutes. This is because if the frequency is too low, it will lead to frequent predictions, which will reduce the prediction effect (and the block size adjustment effect), while too high a frequency will lead to untimely predictions, which will lose the meaning of block size adjustment.

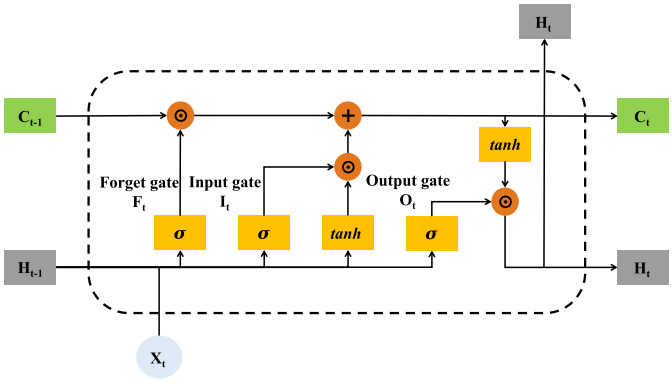


Fig. 3. LSTM cell architecture.

LSTM employs a gating mechanism to save and transmit information, so that it can effectively model long-time series, which is currently the most commonly used method for time series forecasting problems. The internal gating mechanism of each LSTM neuron has a forget gate F, an input gate I, and an output gate O, as is shown in Fig. 3. The input gate retains the current input at a certain proportion. The forget gate controls the ratio of the information transmitted from the upper neuron to the current neuron. The output gate selectively outputs the information contained in the cell. The updated formulas for LSTM neurons are expressed as

$$\begin{cases} \mathrm{I}_t = \sigma\left(\mathrm{X}_t\ \mathrm{W}_{xi} + \mathrm{H}_{t-1}\ \mathrm{W}_{hi} + \mathrm{b}_i\right) \\ \mathrm{F}_t = \sigma\left(\mathrm{X}_t\ \mathrm{W}_{xf} + \mathrm{H}_{t-1}\ \mathrm{W}_{hf} + \mathrm{b}_f\right) \\ \mathrm{O}_t = \sigma\left(\mathrm{X}_t\ \mathrm{W}_{xo} + \mathrm{H}_{t-1}\ \mathrm{W}_{ho} + \mathrm{b}_o\right) \\ \tilde{\mathrm{C}}_t = \tanh\left(\mathrm{X}_t\ \mathrm{W}_{xc} + \mathrm{H}_{t-1}\ \mathrm{W}_{hc} + \mathrm{b}_c\right) \\ \mathrm{C}_t = \mathrm{F}_t \odot \mathrm{C}_{t-1} + \mathrm{I}_t \odot \tilde{\mathrm{C}}_t \\ \mathrm{H}_t = \mathrm{O}_t \odot \tanh\left(\mathrm{C}_t\right) \end{cases} \tag{7}$$

Note that we should use a fully connected layer to map the hidden state H_t to the final predicted output X_{t+1}, which can be expressed as

$$\mathrm{X}_{t+1} = \mathrm{H}_t\ \mathrm{W}_{hx} + \mathrm{b}_x \tag{8}$$

B. The Blockchain Performance Scoring Model

We model the impact of arrival rate and block size on blockchain performance via MLP which is shown in Fig. 4. The input layer is composed of two features, arrival rate and block size. These features are mapped to the output layer after passing through multiple hidden layers. The ground truth of the output layer is either latency or throughput so we need to train two MLP models separately. As an example, the MLP formula with only one hidden layer can be expressed as

$$\begin{cases} \mathrm{H} = \sigma\left(\mathrm{X}\mathrm{W}^{(1)} + \mathrm{b}^{(1)}\right) \\ \mathrm{O} = \mathrm{H}\mathrm{W}^{(2)} + \mathrm{b}^{(2)} \end{cases} \tag{9}$$

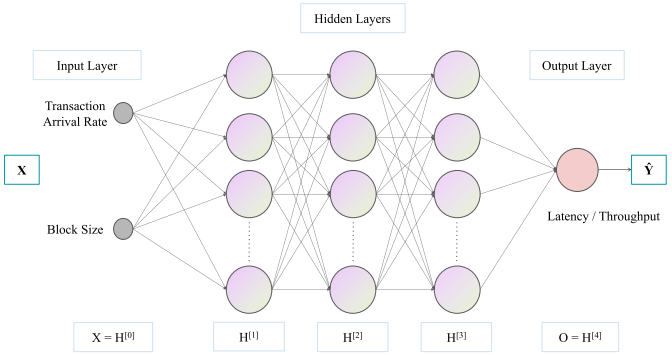


Fig. 4. Multilayer perceptron neural network structure.

where X and H denote the input features and output hidden state, $W^{(1)}$, $W^{(2)}$, $b^{(1)}$, and $b^{(2)}$ are learnable parameters, σ denotes the sigmoid activation, O represents the output of MLP.

Finally, we get different latency and throughput pairs by feeding the predicted number of transaction arrival rate at the next time step together with different block sizes into the above trained MLP model. To determine the optimal block size at the next time step, we define a score function that can be expressed as

$$score = \alpha \cdot score_{latency} + \beta \cdot score_{throughput} \tag{10}$$

where $score_{latency}$ and $score_{throughput}$ are the score of latency, and the socre of throughput, respectively, α (0, 1) is the weight factor for latency, $\beta = 1 - -\alpha$, and which is the weight factor for throughput. In practical scenes, the two weights can be adjusted as needed.

$score_{latency}$ can be expressed as

$$score_{latency} = \frac{latency - latency_{max}}{latency_{min} - latency_{max}} \tag{11}$$

where $latency$ represents the latency corresponding to a certain block size, $latency_{max}$ represents the maximum value of the latency corresponding to the size of all optional blocks, and $latency_{min}$ represents the minimum value of the latency corresponding to the size of all optional blocks.

$score_{throughput}$ can be expressed as

$$score_{throughput} = \frac{throughput - throughput_{min}}{throughput_{max} - throughput_{min}} \tag{12}$$

where $throughput$ represents the throughput corresponding to a certain block size, $throughput_{min}$ represents the minimum value of the throughput corresponding to the size of all optional blocks, and $throughput_{max}$ represents the maximum value of the throughput corresponding to the size of all optional blocks.

The block size corresponding to the highest score is the optimal block size corresponding to the next time slice. The specific process is shown in Algorithm 2. We divide the execution flow of Algorithm 2 into 3 steps, details are given as follows.

1) Predict the transaction arrival rate: Predict the transaction arrival rate after 5 minutes based on the previous transaction arrival rate, as shown in line 10 of the algorithm.

Algorithm 2 Dynamic Adjustment Method of Block Size

1: **Input:**
2: The transaction arrival rate of last $T = 12$ time steps: $X_T = \{x_{t-T+1}, \ldots, x_t\}$;
3: The trained LSTM model for predicting transaction arrival rate $X_P = \{x_{t+1}, \ldots, x_{t+P}\}$ in the next $P = 1$ time steps: LSTM;
4: The trained MLP model for predicting latency: MLP_1;
5: The trained MLP model for predicting throughput: MLP_2;
6: The set of block sizes that can be selected: $B_S = \{B_1, \ldots, B_n\}$;
7: The weight of the throughput score: α;
8: The weight of the latency score: β.
9: **Procedures:**
10: $X_P = \textbf{LSTM}(X_T)$
11: $score_{best} = 0$
12: **foreach** $(B : BS)$
13: $score_{latency} = \textbf{MLP}_1\ (X_P, B)$
14: $score_{throughtput} = \textbf{MLP}_2\ (X_P, B)$
15: $score = \alpha \cdot score_{latency} + \beta \cdot score_{throughput}$
16: **if** $score > score_{best}$ **then**
17: $score_{best} = score$
18: $B_{best} = B$
19: **end if**
20: **end foreach**
21: Resize the block size to B_{best};
22: **end**

2) Calculate the score: Based on the trained blockchain performance model, the performance of each block size is scored and the optimal block size is selected, as shown in lines 11-19 of the algorithm.
3) Adjust block size: The block size is adjusted to the optimal block size, as shown in line 21 of the algorithm.

The time complexity of Algorithm 2 mainly depends on two parts.

1) The training time complexity of LSTM. As shown in Eq. (7), since the LSTM unit contains 4 sets of parameters corresponding to the input gate, output gate, forget gate and candidate states. In addition, the parameters W and b contained in the fully connected layer as shown in Eq. (8) are also considered. In summary, the complexity of this part is $O(4(n*m+n^2+n)+(n+1)) = O(n^2)$, where n is the hidden size, and m is the input size $= 1$.
2) The time complexity of the scoring model. As shown in Eq. (9), the MLP network contains parameters W and b related to the input size, the number of layers in the hidden layer (set to 2 in this paper), the number of cells in the hidden layer and the output size. In summary, the complexity of this part is $O(N*2*(p*i+i+i*j+j+j*q+q)) = O(N*i*j)$, where N is the number of block size sets, p is input size $= 2$, q is output size $= 1$, i and j are the number of cells in the first hidden layer and the number of cells in the second hidden layer, respectively.

In summary, the time complexity of Algorithm 2 is $O(n^2 + N*i*j)$.

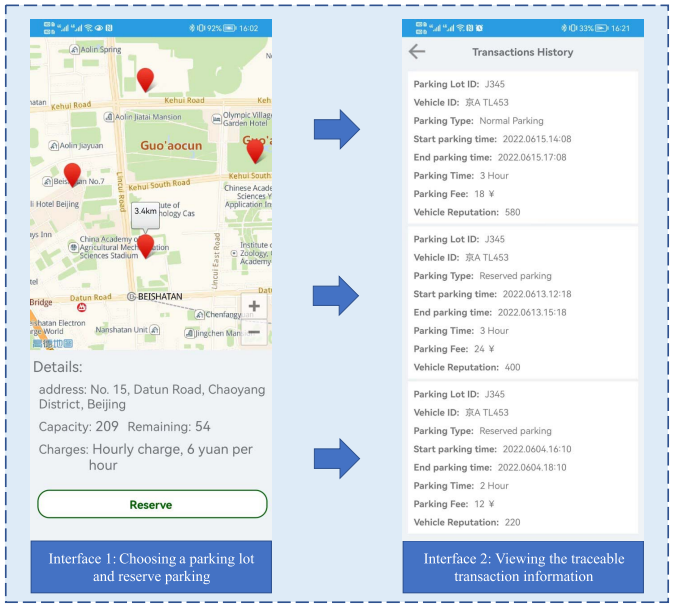


Fig. 5. This prototype system for parking reservations.

TABLE II
PARAMETER SETTINGS OF RESERVATION-BASED PARKING MODE

Parameter	Value
R_{max}	10 km
S_{ave}	500 m/s
T_D	3 hours

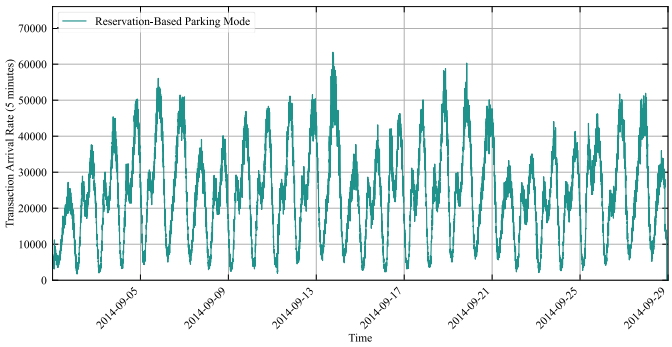


Fig. 6. The transaction arrival rate of reservation-based parking mode.

V. DEPLOYMENT

We deploy Hyperledger Fabric 2.3 on MacBook Pro (64-bit Apple M1 Pro chip and 16GB RAM), and we also deploy a Linux virtual machine on the Macbook Pro with 2 processors and 4GB RAM and installed the ubuntu 20.04 operation system.

Hyperledger Fabric is one of the most popular consortium chain platforms. The transaction throughput of Hyperledger Fabric is several times that of Bitcoin and Ethereum [46], and it has a higher degree of flexibility. Therefore, it is more suitable for high-throughput parking scenes.

We set up 3 Orderer nodes and 5 Peer nodes on Linux virtual machine. In Hyperledger, the Orderer node is responsible for sorting transactions in the network over some time to ensure the order of transactions, and the Peer nodes are responsible for verifying the correctness of transactions and storing the ledger of the blockchain. We set the number of both types of nodes to be odd, which can ensure the stability of the blockchain system as much as possible.

We also implement a prototype system for parking reservations, as shown in Fig. 5, which will be the subject of further research in the future.

VI. EVALUATION AND ANALYSIS

In this section, we evaluate the efficiency of the reservation-based parking mode, the impact of the parking behavior management based on reputation value, and the performance of the dynamic adjustment of block size based on DL methods. In additional, We will also conduct a qualitative analysis of related work.

We use a workstation (64-bit Intel Core i7-11700 2.5GHz CPU, NVIDIA Geforce RTX 3070 GPU, 16GB RAM, and a Windows operating system) to finish the experiment, evaluate and analyze it. The integrated development environment (IDE) is Pycharm, the programming language version is Python 3.6, and the DL framework is PyTorch 1.7.1. These specific experimental details will be described in detail subsequently.

A. The Efficiency of Reservation-Based Parking Mode

To verify the efficiency of the reservation-based parking mode, we use the Algorithm 1 to conduct simulation experiments on real dataset to obtain the result of the transaction arrival rate. In additional, we compare and analyze the time efficiency of the reservation-based parking mode and the regular mode.

We use 1 real-world dataset (Uber Pickups in New York City,[1] this dataset contains data on NYC Uber taxis in 2014). This real-world dataset is preprocessed using the following steps:

1) We treat each request in this dataset as a parking request and randomly scale it up by a factor of 100 (in reality, this number is probably still smaller than the actual situation in New York City).
2) According to a report given by the City of New York, there are between three and five million parking spaces in the city [47]. Therefore, we set up 4 million parking spaces (simulate a shortage of parking spaces), and to avoid as much complexity as possible, we spread them evenly over 1,600 parking lots.
3) We distribute the 1600 parking lots equally according to the location distribution (the range of latitude is [40.64, 40.84] and the range of longitude is [-74.03, -73.83]) of this dataset.

We simulate parking according to the steps in Algorithm I for every parking request in this dataset. The experimental parameter settings are shown in Table II.

The experimental results are shown in Fig. 6. Our proposed reservation-based parking algorithm can efficiently process

[1]https://www.kaggle.com/fivethirtyeight/uber-pickups-in-new-york-city?select=uber-raw-data-sep14.csv

TABLE III
PARAMETER SETTINGS OF REPUTATION MECHANISM

Parameter	Value
R_i^I	100
TS	[60, 180] minutes
TF	[1, 30] minutes
C	1250
ω	5
$\beta_1, \beta_2, \beta_3$	5, 2, 1
λ	-20

parking requests and complete simulated parking. Corresponding to the real-world parking situation, the transaction arrival rate reaches the peak of the day during both the evening peaks.

Compared with the regular parking mode, the reservation parking mode can save more time in finding a parking space. This is because drivers can choose their desired parking lot (parking space) and parking through the App and other means, instead of blindly searching for it after arriving at the parking lot (especially during the peak hours when such a search is likely to be pointless).

B. The Effect of Reservation Parking Behavior Management Based on Reputation Mechanism

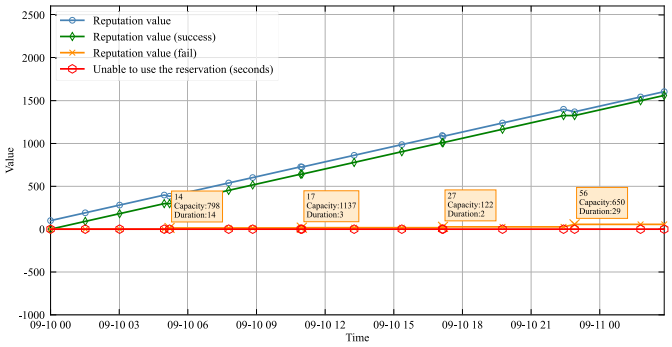
In this part, the experimental parameters are shown in Table III, and we set different parameters to further demonstrate the effect of the proposed reputation mechanism. We conduct simulation experiments on the following three situations:

1) The Change in Reputation Value of Honest Nodes When They Perform Occasional Failure Behaviors: The experimental results are shown in Fig. 7(a). When node i performs the failure behavior for the first time, C_i^1 is 798 and TF_i^1 is 14. Therefore, from Eq. (3) and Eq. (4), it follows that PF_i^j is 1 and R_i^F is 14. When node i performs the second failure behavior, C_i^2 is 1137 and TF_i^2 is 3, so PF_i^2 is 1 and R_i^F is 17 (14 + 1 * 3).

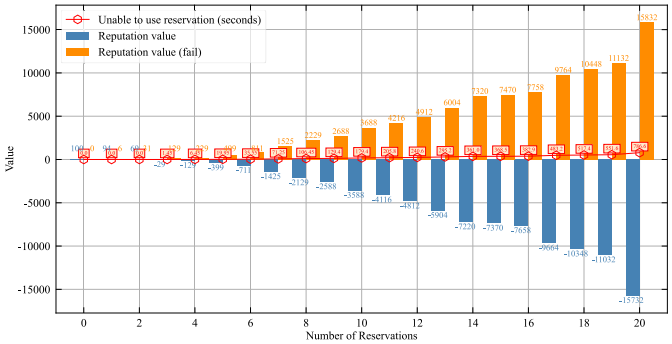
For honest nodes, their occasional conduct of failure does not have a significant impact on the reputation value and is not subject to the penalty of being restricted from parking reservation. Taking node i as an example, it can be known from Eq. (3) that the weights of success and failure behaviors performed by node i are the same, so the reputation value of node i is more stable when the number of failure behaviors of node i is less than its number of success behaviors (unless its number of consecutive failure behaviors is greater than or equal to ω).

2) The Change of Reputation Value When a Malicious Node Performs Frequent Failure Behaviors: As shown in Fig. 7(b). When node i performs the 4^{th} malicious behavior, R_i^F is 229, then R_i is -129. From Eq. (5), node i will be restricted to reserve 6.45 (-129 / -20) seconds. When node i performs the 20^{th} malicious behavior, R_i^F is 15832, then R_i is -15732, and from Eq. (5), node i will be restricted to reserve for 786.6 seconds.

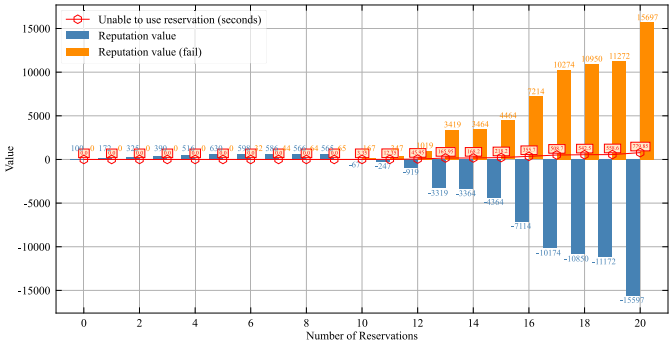
For a malicious node, its reputation value will be affected more and more obvious when it performs frequent failure behaviors, and it will be severely punished at the same time.



(a) When honest node i perform occasionally failure behaviors.



(b) When malicious node i perform frequent failure behaviors.

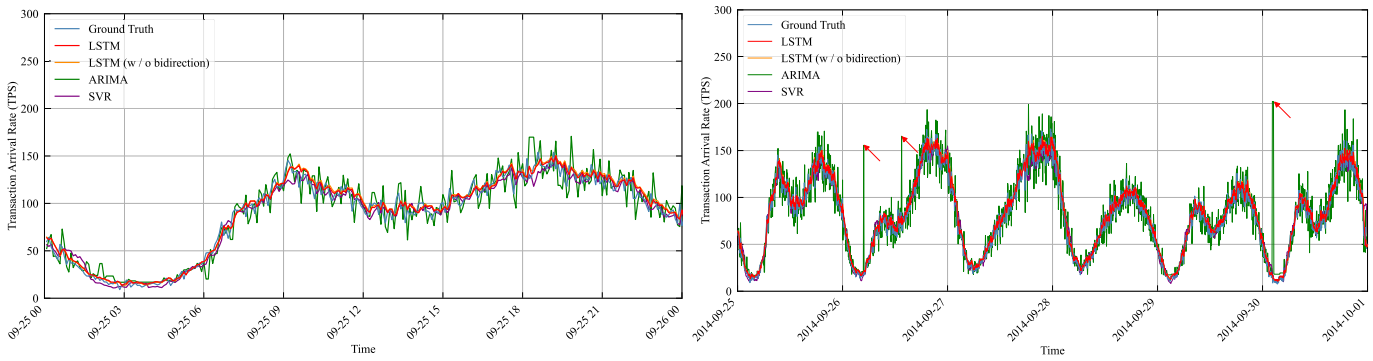


(c) When honest node i suddenly turns into a malicious node.

Fig. 7. The change of reputation value.

Taking node i as an example, from Eq. (3) and Eq. (5), we can know that the weight of failure behavior performed by malicious node i is much greater than that of success behavior. Therefore, once node i becomes a malicious node, it will be difficult to recover back to an honest node (unless it performs multiple success behaviors in a row).

3) The Change of Reputation Value When an Honest Node Suddenly Turns Into a Malicious Node: As shown in Fig. 7(c). After node i performs a large number of success behaviors, it starts to perform malicious behaviors. After node i performs the 5^{th} malicious behavior, even though R_i is still greater than 0, R_i^S is reset to 0 because the number of consecutive failure behaviors of node i reaches ω, while it will be penalized by being restricted from making reservations. When node i performs the 5^{th} malicious behavior, R_i is -67, so it is restricted to reserve for 3.35 (-67 / -20) seconds. When node i performs the 15^{th} malicious behavior, R_i is -15597, so it is restricted to reserve for 779.85 seconds.



(a) 1 day. (b) 6 days.

Fig. 8. The prediction of transaction arrival rate based on LSTM, LSTM (w/o Bidirection), ARIMA, and SVR.

For an honest node, when it suddenly performs malicious behaviors frequently (when the number of consecutive failure behaviors reachs ω), it will be quickly regarded as a malicious node (even if its reputation value is very high), and at the same time it will be severely punished. Which can effectively prevent the extreme case of an honest node suddenly becomes a malicious node.

Therefore, our proposed reputation mechanism can effectively curb the malicious reservation behavior while avoiding affecting the vehicles with normal reservations. By considering various situations, the reputation model has better feasibility and applicability.

C. The Performance of Block Size Dynamic Adjustment

In this part, we complete the following 3 experiments.

1) Prediction of Short-Term Transaction Sending Rate Based on LSTM, LSTM (w/o Bidirection), ARIMA, and SVR: We calculate the transaction arrival rate per minute by conducting experiments using the previously mentioned dataset expanded by a factor of 100. We aggregate the original 1-minute interval data into 5-minute intervals, thus containing 288 records per day for a total of 8640 records. In addition, the input data is mapped between [0, 1] using min-max normalization. We use 70% of the data for training, 20% of the data for testing, and the remaining 10% of the data for validating.

We take the mean squared error (MSE) as the loss function of our model for training. We adapt the commonly used mean absolute error (MAE), mean absolute percentage error (MAPE) and root mean square error (RMSE) as evaluation metrics for regression prediction.

We perform a grid search strategy on the validation set to locate the best hyperparameters. We use the Adam optimizer for training. The learning rate is set to 0.001. The hidden state dimension of LSTM is set to 128, and the hidden layer is set to 2 layers. The batch size is set to 64. Early stopping is used to avoid overfitting. As mentioned earlier, we use 60 minutes as the historical time window, that is, 12 observed data points ($T = 12$) to predict the transaction arrival rate of the parking lot for the next 5 minutes ($P = 1$).

To show the effectiveness of LSTM, we compare LSTM with the following 3 models. ARIMA: Auto-Regressive Integrated Moving Average Model, which is a statistical time series prediction classical model; Support Vector Regression (SVR): SVR is another classical time series prediction model which uses linear support vector machine for the regression task; LSTM (w / o bidirection): LSTM network without bidirection.

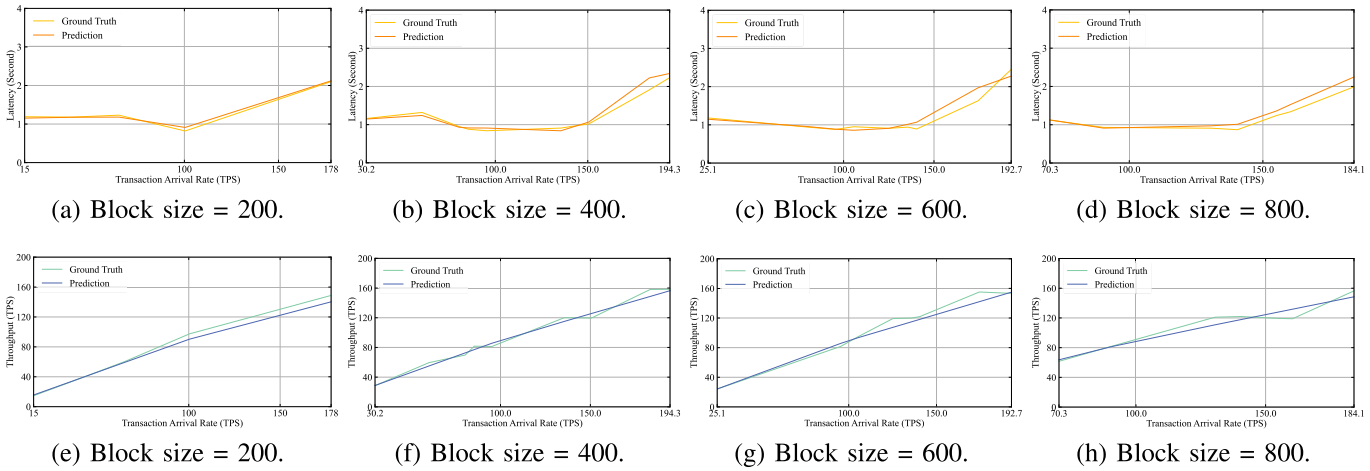
TABLE IV
EXPERIMENT RESULTS

Model	MAE	RMSE	MAPE (%)
ARIMA	3831.86	15381.00	22.96
SVR	1934.40	2580.11	10.07
LSTM (w / o bidirectional)	1793.55	2327.17	8.93
LSTM	1747.98	2260.08	8.99

Note: LSTM (i.e., LSTM Univariate, in this paper, we only use transaction arrival rate as variable.)

Table IV compares the prediction performance of LSTM with 3 different models. LSTM achieves the minimum prediction error, because compared with the statistical prediction model and the traditional machine learning model, LSTM can effectively capture short-term and long-term temporal dependencies.

To better demonstrate the superiority of LSTM based transaction arrival rate prediction model, we visually compare the prediction results of LSTM and the above three models. The experimental results are shown in Fig. 8. Fig. 8. (a) shows the visualization of the prediction results and ground truth for 1 days in September 25. Fig. 8. (b) shows the visualization of the prediction results and ground truth for 6 days from September 25 to September 30. We have the following observations. First, As indicated by the red arrow in Fig. 8. (b), extreme outliers will appear in ARIMA prediction results, which indicates the unreliability of ARIMA model. Moreover, ARIMA prediction results fluctuate greatly, which also indicates the instability of ARIMA. Second, compared with the traditional machine learning model SVR, the LSTM prediction results fit the ground truth better. Third, when there is a small fluctuation in the transaction arrival rate, the prediction value generated by the LSTM model is smoother, which reflects the robustness of the model. Forth, the prediction results of the model are also accurate for sharp changes in the transaction arrival rate, which reflects the flexibility of the LSTM model.



(a) Block size = 200. (b) Block size = 400. (c) Block size = 600. (d) Block size = 800.

(e) Block size = 200. (f) Block size = 400. (g) Block size = 600. (h) Block size = 800.

Fig. 9. The prediction effect of blockchain performance prediction model.

2) Modeling the Impact of Transaction Arrival Rate and Block Size on Blockchain Performance Based on MLP: We set the transaction arrival rate from 10 to 200 with an interval of 5, and block sizes from 10 to 800 for each transaction arrival rate above, with an interval of 10. According to the above settings for the transaction arrival rate and block size, we obtain the corresponding delay and throughput through the performance test of the blockchain, thus obtaining 3120 pieces of data. In addition, transaction arrival rate and block size are mapped between [0, 1] using min-max normalization. We use 80% of the data for training, and 20% of the data for testing.

We use MLP to perform the training of the blockchain performance model. In which we use early stop to determine the number of epochs for model training. When the number of epochs is 5000, the loss function of the model almost stops decreasing and the model can be trained in roughly 5 minutes.

We take MSE as the loss function of our model for training. We adapt the commonly used MAE, MAPE, RMSE as evaluation metric for the regression prediction.

In both MLPs trained separately for latency and throughput, we use the Adam optimizer. The learning rate is set to 0.001. The hidden layer is set to 2 layers. The dimension of the first hidden layer is set to 64, and the dimension of the second hidden layer is set to 8. The batch size is set to 32. dropout is used to avoid overfitting. The experimental results of our trained blockchain performance model are shown in Table V.

Fig. 9. shows a visualization of the prediction results and ground truth of the blockchain performance prediction model on the test set. Among them, Fig. 9. (a)-(d) is the prediction of latency, and Fig. 9. (e)-(h) is the prediction of throughput. We have the following observation that given the inputs of transaction arrival rate and block size, the blockchain performance prediction model can predict latency and throughput with quite accuracy.

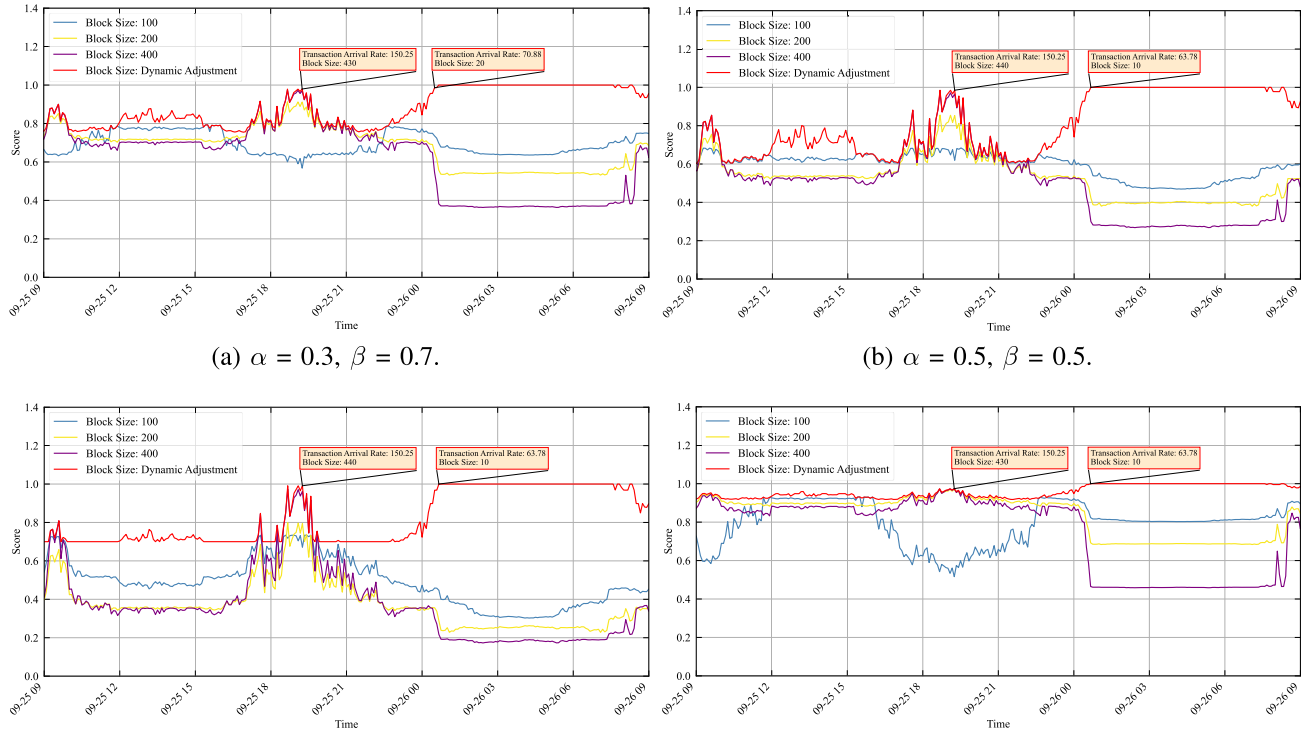
3) Performance of DL-Based Dynamic Block Size Adjustment Method: The experimental results are shown in Fig. 10. Depending on the arrival rate of transactions at different times (basically equivalent to parking traffic), different block sizes correspond to very different scores. In some periods, there are even significant differences, for example, at 1:00 a.m. on September 26 in Fig. 10 (c), the block size of 400 corresponds to a score of 0.2, while the dynamically selected optimal block size of 10 corresponds to a score of 0.99. Setting different block sizes will have a significant impact on the performance of the blockchain, which also illustrates the necessity of choosing the optimal block size.

We set different weights to accurately evaluate the effectiveness of our proposed method. Obviously, our method can select the optimal block size based on the real-time transaction arrival rate compared to a fixed block size, resulting in real-time optimal blockchain performance.

The difference in weights will significantly affect the blockchain performance scores. In some latency-sensitive scenarios (e.g., banking transactions), latency is more important compared to throughput, while in some throughput-sensitive scenarios (e.g., online shopping sites), the opposite is true. Therefore, in real scenes, the weights need to be set according to the actual needs.

We also select 3 related work to further illustrate the effectiveness and scalability of our proposed method. The experimental parameters of these 3 related work are shown in Table V. To ensure the experimental environment is as uniform and fair as possible (even so, the differences caused by the computational performance of the devices still cannot be excluded), we re-perform the blockchain performance experiments and train the new blockchain performance models in our local environment according to the parameters they set. The accuracy of these three blockchain performance models is still measured using MAE, RMSE and MAPE. The training results are shown in Table V.

The experimental results are shown in Fig. 11, to ensure the fairness of the experiment, we set the collection of selectable block size all to [10, 20, …, 200] (even though some block size may achieve better performance, as in Fig. 10). Our method can effectively improve their blockchain performance based on the 3 related work. As shown in Fig. 11 (a), we set α to 0.9 and β to 0.1 (with latency as the main weight), and the dynamically chosen block size can make the latency reach the optimal value, in most cases, the chosen block size has the lowest latency (all cases if we set α to 1, and β to 0).



(a) $\alpha = 0.3$, $\beta = 0.7$.

(b) $\alpha = 0.5$, $\beta = 0.5$.

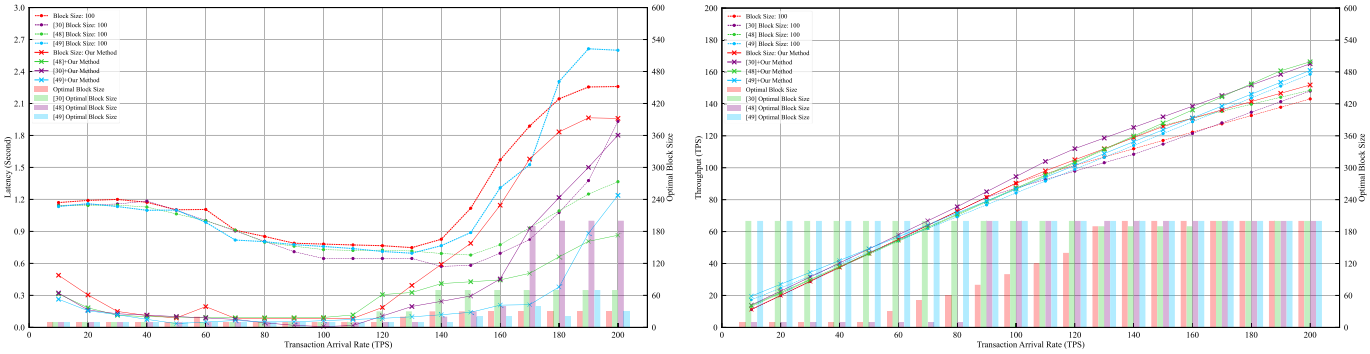
(c) $\alpha = 0.7$, $\beta = 0.3$.

(d) $\alpha = 0.1$, $\beta = 0.9$ (throughput is the main indicator).

Fig. 10. Dynamically selected optimal block size by score model.

TABLE V
EXPERIMENT RESULTS

Work	Blockchain Platform	Orderers	Peers	Endorsement Policies	Transaction Arrival Rate	Block Size	Model	MAE	RMSE	MAPE (%)
[30], 2021	Hyperledger Fabric	3	4	*OR*	[10, 20, ..., 200]	[10, 20, ..., 200]	MLP_1	0.139	0.286	12.99
							MLP_2	4.226	6.360	5.298
[48], 2022	Hyperledger Fabric	1	4	*OR*	[10, 20, ..., 200]	[10, 20, ..., 200]	MLP_1	0.104	0.168	10.66
							MLP_2	4.339	5.998	6.390
[49], 2021	Hyperledger Fabric	1	4	*AND*	[10, 20, ..., 200]	[10, 20, ..., 200]	MLP_1	0.116	0.253	11.18
							MLP_2	6.896	9.258	11.77
Ours	Hyperledger Fabric	3	5	*MAJORITY*	[10, 15, ..., 200]	[10, 20, ..., 800]	MLP_1	0.087	0.149	6.918
							MLP_2	3.450	4.971	4.218



(a) $\alpha = 0.9$, $\beta = 0.1$ (latency is the main indicator).

(b) $\alpha = 0.1$, $\beta = 0.9$ (throughput is the main indicator).

Fig. 11. The optimization effect of the proposed block size dynamic adjustment method.

As shown in Fig. 11 (b), we set α to 0.1 and β to 0.9 (with throughput as the main weight), and the dynamically chosen block size enables the throughput to reach the optimal value, in all cases, the chosen block size has the highest throughput.

TABLE VI
THE COMPARISONS WITH RELATED WORK

Works	Based-Blockchain	Blockchain Platform	Parking Reservation	Reputation Mechanism	Blockchain Performance	Security
[33], 2019	✓	Ethereum	-	-	Low	✓
[34], 2020	✓	Hyperledger Fabric	-	-	High	✓
[35], 2020	✓	-	-	-	High	✓
[18], 2021	-	-	✓	-	-	✓
[50], 2022	-	-	✓	-	-	-
Ours	✓	Hyperledger Fabric	✓	✓	**High and dynamic optimization**	✓

Note: ✓is "Yes" (related to this content), - is "No" (not related to this content or can not be realized).

The experimental results show that the block size will have a significant impact on the blockchain performance, which is accompanied by a certain pattern as the block size changes. In conclusion, in a practical scenario, the maintainer of the blockchain can further adjust the values of the two weights according to realistic needs (e.g., when only throughput needs to be considered, α can be set to 0 and β to 1).

This dataset[2] for this subsection we have open sourced on Kaggle, and the code for this subsection we have open sourced on GitHub. Researchers can conduct further studies with the dataset and the code[3] we provide.

D. System Analysis

We compare the proposed scheme with related work and perform a qualitative analysis to illustrate the innovation of our work.

We choose 5 related work, analyze and compare them from multiple aspects, as shown in Table VI. [33] presented a framework for a blockchain-based parking management system that aims to protect the privacy of users, but they used Ethereum as the blockchain platform and were, therefore, less efficient. Both [34] and [35] proposed specific parking framework, their efforts are focused on the sharing of private parking spaces. However, they did not take into account the research work on parking reservations as well as reputation mechanisms. [18] presented a learning automata and reservation-based smart parking system. Finally, [50] presented an efficient parking reservation framework that accurately predicts parking loads and effectively relieves parking loads in parking lots. However, they did not consider the case of malicious reservations or the security of the system.

In short, compared to the existing research work, our proposed work focuses on the overall solution of the parking reservation system and we consider the parking efficiency, malicious reservations, system efficiency, and security. Such a holistic solution is not yet considered in other current efforts, so our work can be used in other similar scenes.

In particular, to the best of our knowledge, our proposed deep learning-based method for dynamic block size adjustment, is the first of its kind. This method can be used in many blockchain performance-sensitive scenes.

[2]https://www.kaggle.com/datasets/loveffc/blockchain-performance
[3]https://github.com/JiShuWang/BPR

VII. CONCLUSION

In this work, to improve the efficiency of parking and the security of the parking system, we propose a blockchain-based reserved parking scheme. The reputation mechanism proposed by us can effectively curb the occurrence of malicious reservation behavior and ensure the efficiency of reserved parking. Our proposed DL-based dynamic block size adjustment method and scoring model realize the selection of the optimal block size. Many experiments and evaluation results show that the system has good performance. Our proposed block size dynamic adjustment method is applicable not only to Hyperledger Fabric but also to mainstream blockchain platforms, especially for blockchain performance-sensitive scenarios. In conclusion, our work is not only applicable to the parking lot scene, but also to many similar scenarios. However, our proposed scheme has some shortcomings, including the development of a parking reservation system that is still in the prototype stage.

In future work, we will further investigate the blockchain performance model and parking reservation system. First of all, more factors affecting blockchain performance (e.g., endorsement strategy, number of nodes, block batch processing time, transaction size) will be considered and deep learning models with better results will be selected for training to further improve the accuracy and robustness of blockchain performance models. Secondly, we will use Markov model and game theory to explore and study the parking model more deeply and scientifically, with a view to establishing a systematic model that affects drivers' parking decisions.

REFERENCES

[1] M. A. Merzoug, A. Mostefaoui, G. Gianini, and E. Damiani, "Smart connected parking lots based on secured multimedia IoT devices," *Computing*, vol. 103, no. 6, pp. 1143–1164, Jun. 2021.
[2] Z. Xie, X. Wu, J. Guo, and Z. Zhan, "Parking lot allocation model considering conversion between dynamic and static traffic," *J. Intell. Fuzzy Syst.*, vol. 41, no. 4, pp. 5207–5217, Nov. 2021.
[3] Z. Liu, D. Li, Y. Yang, X. Chen, X. Lv, and X. Li, "Design and implementation of the optimization algorithm in the layout of parking lot guidance," *Wireless Commun. Mobile Comput.*, vol. 2021, Apr. 2021, Art. no. 6639558.
[4] U. TODAY. *Drivers Spend an Average of 17 Hours a Year Searching for Parking Spots*. Accessed: Jul. 2017. [Online]. Available: https://www.usatoday.com/story/money/2017/07/12/parking-pain-causes-financial-and-personal-strain/467637001/
[5] H. Gao, Q. Yun, R. Ran, and J. Ma, "Smartphone-based parking guidance algorithm and implementation," *J. Intell. Transp. Syst.*, vol. 25, no. 4, pp. 412–422, Jul. 2021.
[6] A. O. Kotb, Y.-C. Shen, and Y. Huang, "Smart parking guidance, monitoring and reservations: A review," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 2, pp. 6–16, Summer 2017.

[7] T.-C. Tsai and Y. Chen, "An IoT based parking recommendation system considering distance and parking lot flow," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2021, pp. 978–983.

[8] R. K. Kasera and T. Acharjee, "A smart indoor parking system," *Social Netw. Comput. Sci.*, vol. 3, no. 1, p. 9, Jan. 2022.

[9] Y. Atif, S. Kharrazi, D. Jianguo, and S. F. Andler, "Internet of Things data analytics for parking availability prediction and guidance," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 5, p. e3862, May 2020.

[10] J. Liu, J. Wu, and L. Sun, "Control method of urban intelligent parking guidance system based on Internet of Things," *Comput. Commun.*, vol. 153, pp. 279–285, Mar. 2020.

[11] D. Zhao, Z. Cao, C. Ju, D. Zhang, and H. Ma, "D2Park: Diversified demand-aware on-street parking guidance," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 4, no. 4, p. 163, 2020.

[12] O. Tran Thi Kim, N. H. Tran, C. Pham, T. LeAnh, M. T. Thai, and C. S. Hong, "Parking assignment: Minimizing parking expenses and balancing parking demand among multiple parking lots," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1320–1331, Jul. 2020.

[13] L. Zhu, M. Li, Z. Zhang, and Z. Qin, "ASAP: An anonymous smart-parking and payment scheme in vehicular networks," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 4, pp. 703–715, Jul. 2020.

[14] Y. Duan, N. Chen, S. Shen, P. Zhang, Y. Qu, and S. Yu, "FDSA-STG: Fully dynamic self-attention spatio-temporal graph networks for intelligent traffic flow prediction," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 9250–9260, Sep. 2022.

[15] J. Parmar, P. Das, and S. M. Dave, "Study on demand and characteristics of parking system in urban areas: A review," *J. Traffic Transp. Eng.*, vol. 7, no. 1, pp. 111–124, Feb. 2020.

[16] P. Zhao, H. Guan, P. Wang, and H. Yan, "Evaluation of environmental benefits caused by reservation-based shared parking: A case study of Beijing, China," *IEEE Access*, vol. 9, pp. 3744–3751, 2021.

[17] A. Waheed, P. V. Krishna, G. J. B. Sadoun, and M. Obaidat, "Learning automata and reservation based secure smart parking system: Methodology and simulation analysis," *Simul. Model. Pract. Theory*, vol. 106, Jan. 2021, Art. no. 102205.

[18] R. Huang, C. Lu, X. Lin, and X. Shen, "Secure automated valet parking: A privacy-preserving reservation scheme for autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11169–11180, Nov. 2018.

[19] S. Jain et al., "Blockchain and autonomous vehicles: Recent advances and future directions," *IEEE Access*, vol. 9, pp. 130264–130328, 2021.

[20] L. Campanile, M. Iacono, F. Marulli, and M. Mastroianni, "Designing a GDPR compliant blockchain-based IoV distributed information tracking system," *Inf. Process. Manage.*, vol. 58, no. 3, May 2021, Art. no. 102511.

[21] I. Aliyu, M. C. Feliciano, S. van Engelenburg, D. O. Kim, and C. G. Lim, "A blockchain-based federated forest for SDN-enabled in-vehicle network intrusion detection system," *IEEE Access*, vol. 9, pp. 102593–102608, 2021.

[22] D. Chulerttiyawong and A. Jamalipour, "A blockchain assisted vehicular pseudonym issuance and management system for conditional privacy enhancement," *IEEE Access*, vol. 9, pp. 127305–127319, 2021.

[23] X. Huang, P. Li, R. Yu, Y. Wu, K. Xie, and S. Xie, "FedParking: A federated learning based parking space estimation with parked vehicle assisted edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 9355–9368, Sep. 2021.

[24] Z. Li, M. Alazab, S. Garg, and M. S. Hossain, "PriParkRec: Privacy-preserving decentralized parking recommendation service," *IEEE Trans. Veh. Technol.*, vol. 70, no. 5, pp. 4037–4050, May 2021.

[25] S. Ananthanarayanan Bragadeesh and A. Umamakeswari, "Secured vehicle life cycle tracking using blockchain and smart contract," *Comput. Syst. Sci. Eng.*, vol. 41, no. 1, pp. 1–18, 2022.

[26] G. Subramanian and A. S. Thampy, "Implementation of hybrid blockchain in a pre-owned electric vehicle supply chain," *IEEE Access*, vol. 9, pp. 82435–82454, 2021.

[27] S. R. Maskey, S. Badsha, S. Sengupta, and I. Khalil, "Reputation-based miner node selection in blockchain-based vehicular edge computing," *IEEE Consum. Electron. Mag.*, vol. 10, no. 5, pp. 14–22, Sep. 2021.

[28] F. Ayaz, Z. Sheng, D. Tian, and Y. L. Guan, "A proof-of-quality-factor (PoQF)-based blockchain and edge computing for vehicular message dissemination," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2468–2482, Feb. 2021.

[29] L. Cui et al., "A blockchain-based containerized edge computing platform for the internet of vehicles," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2395–2408, Feb. 2021.

[30] J. Wang, R. Zhu, T. Li, F. Gao, Q. Wang, and Q. Xiao, "ETC-oriented efficient and secure blockchain: Credit-based mechanism and evidence framework for vehicle management," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11324–11337, Nov. 2021.

[31] G. Tasseron and K. Martens, "Urban parking space reservation through bottom-up information provision: An agent-based analysis," *Comput., Environ. Urban Syst.*, vol. 64, pp. 30–41, Jul. 2017.

[32] C. Wan, J. Zhang, and D. Huang, "SCPR: Secure crowdsourcing-based parking reservation system," *Secur. Commun. Netw.*, vol. 2017, May 2017, Art. no. 1076419.

[33] J. Hu, D. He, Q. Zhao, and K.-K.-R. Choo, "Parking management: A blockchain-based privacy-preserving system," *IEEE Consum. Electron. Mag.*, vol. 8, no. 4, pp. 45–49, Jul. 2019.

[34] C. Zhang et al., "BSFP: Blockchain-enabled smart parking with fairness, reliability and privacy protection," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6578–6591, Jun. 2020.

[35] L. Wang, X. Lin, E. Zima, and C. Ma, "Towards airbnb-like privacy-enhanced private parking spot sharing based on blockchain," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 2411–2423, Mar. 2020.

[36] M. M. Badr, W. A. Amiri, M. M. Fouda, M. M. E. A. Mahmoud, A. J. Aljohani, and W. Alasmary, "Smart parking system with privacy preservation and reputation management using blockchain," *IEEE Access*, vol. 8, pp. 150823–150843, 2020.

[37] Y. Zhang, C. Li, N. Chen, and P. Zhang, "Intelligent requests orchestration for microservice management based on blockchain in software defined networking: A security guarantee," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2022, pp. 254–259.

[38] S. Rathore, J. H. Park, and H. Chang, "Deep learning and blockchain-empowered security framework for intelligent 5G-enabled IoT," *IEEE Access*, vol. 9, pp. 90075–90083, 2021.

[39] S. Wang, S. Sun, X. Wang, Z. Ning, and J. J. P. C. Rodrigues, "Secure crowdsensing in 5G internet of vehicles: When deep reinforcement learning meets blockchain," *IEEE Consum. Electron. Mag.*, vol. 10, no. 5, pp. 72–81, Sep. 2021.

[40] M. Z. Khan, M. U. G. Khan, O. Irshad, and R. Iqbal, "Deep learning and blockchain fusion for detecting driver's behavior in smart vehicles," *Internet Technol. Lett.*, vol. 3, no. 6, Nov. 2020.

[41] Y. Dai, D. Xu, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4312–4324, Apr. 2020.

[42] G. O. Boateng, G. Sun, D. A. Mensah, D. M. Doe, R. Ou, and G. Liu, "Consortium blockchain-based spectrum trading for network slicing in 5G RAN: A multi-agent deep reinforcement learning approach," *IEEE Trans. Mobile Comput.*, early access, Jul. 19, 2022, doi: 10.1109/TMC.2022.3190449.

[43] G. O. Boateng, D. Ayepah-Mensah, D. M. Doe, A. Mohammed, G. Sun, and G. Liu, "Blockchain-enabled resource trading and deep reinforcement learning-based autonomous RAN slicing in 5G," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 1, pp. 216–227, Mar. 2022.

[44] K. L.-M. Ang and J. K. P. Seng, "Embedded intelligence: Platform technologies, device analytics, and smart city applications," *IEEE Internet Things J.*, vol. 8, no. 17, pp. 13165–13182, Sep. 2021.

[45] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[46] C. Gorenflo, S. Lee, L. Golab, and S. Keshav, "FastFabric: Scaling hyperledger fabric to 20 000 transactions per second," *Int. J. Netw. Manage.*, vol. 30, no. 5, Sep. 2020.

[47] S. NYC. *Op-ED: To Break the Car Culture, Reduce the Number of Parking Spots*. Accessed: Aug. 2019. [Online]. Available: https://nyc.streetsblog.org/2019/08/21/op-ed-to-break-the-car-culture-reduce-the-number-of-parking-spots/comment-page-1/

[48] P. Gaba, R. S. Raw, M. A. Mohammed, J. Nedoma, and R. Martinek, "Impact of block data components on the performance of blockchain-based VANET implemented on hyperledger fabric," *IEEE Access*, vol. 10, pp. 71003–71018, 2022.

[49] X. Xu, G. Sun, L. Luo, H. Cao, H. Yu, and A. V. Vasilakos, "Latency performance modeling and analysis for hyperledger fabric blockchain network," *Inf. Process. Manage.*, vol. 58, no. 1, Jan. 2021, Art. no. 102436.

[50] X. Zhang, F. Yuan, Y. Cao, and S. Liu, "Reservation enhanced autonomous valet parking concerning practicality issues," *IEEE Syst. J.*, vol. 16, no. 1, pp. 351–361, Mar. 2022.

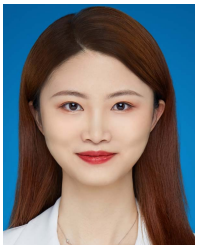
Jishu Wang (Member, IEEE) received the B.E. degree in computer science and technology from Kunming University, Kunming, China, in 2019, and the M.E. degree in software engineering from Yunnan University, Kunming, in 2022, where he is currently pursuing the Ph.D. degree with the School of Information Science and Engineering. His current research interests include blockchain technology, intelligent transportation, and smart grid.

Chao Zhu received the B.E. degree in new energy science and engineering from Hohai University, Nanjing, China, in 2017. He is currently pursuing the M.E. degree with the School of Software, Yunnan University, Kunming, China. His current research interests include spatiotemporal data mining and traffic prediction.

Chen Miao received the B.E. degree in information management and information system from Henan Normal University, Xinxiang, China, in 2019. He is currently pursuing the M.E. degree with the School of Software, Yunnan University, Kunming, China. His current research interests include blockchain technology and intelligent transportation.

Rui Zhu (Member, IEEE) received the Ph.D. degree in software engineering from Yunnan University, Kunming, China, in 2016. He is currently the Head and an Associate Professor of artificial intelligence with the School of Software, Yunnan University. His current research interests include intelligent transportation, blockchain technology, and deep learning.

Xuan Zhang (Member, IEEE) received the B.S. and M.S. degrees in computer science and the Ph.D. degree in system analysis and integration from Yunnan University, Kunming, China. She is currently a Professor with the School of Software, Yunnan University, Yunnan, China. She is also the Core Scientist of the Yunnan Key Laboratory of Software Engineering and the Yunnan Software Engineering Academic Team. She has been a principal investigator for more than 30 national, provincial, and private grants and contracts. She is the author of three books and more than 90 articles. Her research interests include blockchain, knowledge graph (KG), natural language processing (NLP), and recommendation systems.



Yahui Tang received the Ph.D. degree from Yunnan University, Kunming, China, in 2022. She is currently a Lecturer at the School of Software Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China. Her main research interests include process mining, business process, and software process.

Hexiang Huang received the B.E. degree in software engineering from the Hubei University of Automotive Technology, Shiyan, China, in 2020. He is currently pursuing the M.E. degree with the School of Software, Yunnan University, Kunming, China. His current research interests include blockchain security and deep learning.

Chen Gao (Graduate Student Member, IEEE) received the M.E. degree in software engineering from Yunnan University, Kunming, China, where he is currently pursuing the Ph.D. degree with the School of Information Science and Engineering. He published three articles in the field of information extraction. His research interests include natural language processing and knowledge graph, especially information extraction.